DOCUMENT RESUME

ED 261 656 IR 011 805

AUTHOR Riley, Mary S.
TITLE User Understanding.
INSTITUTION California Univ., San Diego, La Jolla. Inst. for Cognitive Science.
SPONS AGENCY Office of Naval Research, Arlington, Va.
REPORT NO ICS-8504
PUB DATE May 85
CONTRACT N00014-84-K-0579; NR-667-538
NOTE 21p.; For related document, see IR 011 804.
PUB TYPE Viewpoints (120) -- Reports - Research/Technical (143)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS Cognitive Development; *Cognitive Mapping; *Cognitive Processes; *Computers; Integrated Activities; Learning Strategies; *Man Machine Systems; Models; *Problem Solving
IDENTIFIERS Cognitive Psychology

ABSTRACT

This document presents hypotheses about how much understanding a user needs to perform skillfully using a computer or a computer program. A framework for characterizing user understanding is presented which includes three criteria for evaluating the representation generated during problem solving: (1) internal coherence--whether the components of the user's knowledge are related in an integrated structure; (2) validity--whether the user's knowledge accurately reflects the behavior of the device; and (3) integration--the extent to which the user's knowledge about a device is tied to other components of a user's knowledge. Each criterion is defined, discussed in detail, and illustrated by examples. The summary reiterates the multi-dimensional quality of understanding and argues that the degree of internal coherence, validity, and integration does not depend on a single aspect of knowledge, but upon several. It is stressed that a user should not be considered as either performing with or without understanding, since it is possible for the user to have acquired some components of knowledge and not others. Lists of 33 references and 41 publications written by people in the Institute for Cognitive Science complete the document. (JB)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED261656

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
✓ This document has been reproduced as received from the person or organization originating it.
☐ Minor changes have been made to improve reproduction quality.
• Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

BEST COPY AVAILABLE

USER UNDERSTANDING

Mary S. Riley

May 1985

ICS Report 8504

INSTITUTE FOR COGNITIVE SCIENCE
UNIVERSITY OF CALIFORNIA, SAN DIEGO LA JOLLA, CALIFORNIA 92093

IR011805

BEST COPY AVAILABLE

USER UNDERSTANDING

Mary S. Riley

May 1985

ICS Report 8504

Institute for Cognitive Science
University of California, San Diego C-015
La Jolla, CA 92093

To be published in D. A. Norman & S. W. Draper (Eds.) (1986), *User centered system design: New perspectives in human-computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates.

The research reported here was conducted under Contract N00014-84-K-0579, NR 667-538 with the Personnel and Training Research Programs of the Office of Naval Research. The views and conclusions contained in this document are those of the author and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsoring agency. Approved for public release; distribution unlimited. Reproduction in whole or in part is permitted for any purpose of the United Stated Government. Requests for reprints should be sent to Mary Riley, Institute for Cognitive Science, C-015; University of California, San Diego; La Jolla, CA 92093.
Copyright © 1985 by Mary S. Riley

Unclassifed
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassifed	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release: distribution unlimited.
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) ICS 8504	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Institute for Cognitive Science 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) C-015 University of California, San Diego La Jolla, CA 92093	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING / SPONSORING ORGANIZATION Personnel & Training Research 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N-00014-84-K-0579
8c. ADDRESS (City, State, and ZIP Code) Code (442 PT) Office of Naval Research 800 North Quincy Street, Arlington VA 22217	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. NR 667-538 / TASK NO. / WORK UNIT ACCESSION NO
11. TITLE (Include Security Classification) User Understanding	
12. PERSONAL AUTHOR(S) Mary S. Riley	
13a. TYPE OF REPORT Technical 13b. TIME COVERED FROM 12-84 TO 3-85	14. DATE OF REPORT (Year, Month, Day) May 1985 15. PAGE COUNT 11
16. SUPPLEMENTARY NOTATION To be published in D. A. Norman & S. W. Draper (Eds.), User centered system design: New perspectives in human-computerinteraction. Hillsdale, NJ: Lawrence Erlbaum Assoc.	
17. COSATI CODES FIELD / GROUP / SUB-GROUP	18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number) Expert-novice users; human-computer interaction; learning; problem solving; understanding
19. ABSTRACT (Continue on reverse if necessary and identify by block number) OVER	
20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code) / 22c. OFFICE SYMBOL

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
unclassified

ABSTRACT

In this chapter I explore some ideas about how much understanding a user needs to perform skillfully using a device. I suggest a framework for characterizing user understanding and discuss the role of understanding in performance and learning. I propose that (a) the level at which a user interacts with a device is determined by the tasks being performed, (b) the device's functions and structures that are understood differ from level to level, and (c) a uniform set of criteria is appropriate for evaluating understanding at any level. The criteria concern three aspects of a user's knowledge about a device:

1. *Coherence* — are the components of the user's knowledge related in an integrated structure?
2. *Validity* — does the user's knowledge reflect the actual behavior of the device?
3. *Integration* — to what extent is the user's knowledge about a device tied to other components of a user's knowledge?

I discuss how Coherence, Validity, and Integration facilitate learning, improve the efficiency, flexibility, and reliability of performance, provide predictive and explanatory power, increase the likelihood that procedures will be remembered or can be regenerated, and enable the transfer of skills.

Contents

A FRAMEWORK FOR CHARACTERIZING UNDERSTANDING 2
WHAT KIND OF UNDERSTANDING DOES A USER NEED? 3
- Internal Coherence 3
- Validity 6
- Integration 8

SUMMARY 9
REFERENCES 9

User Understanding

MARY S. RILEY

My goal in this chapter is to explore how much understanding a user needs to perform skillfully. My interest in the relationship between understanding and skilled performance is currently focused on two areas: basic electronics and human-computer interfaces.

In the study of how people learn basic electronics, I am analyzing how theoretical understanding of physics might influence skilled performance in analyzing and troubleshooting electronic circuits. A traditional assumption is that formal physics is necessary for skilled task performance. Yet many skilled electronics technicians probably do not use formal physics to perform these tasks, and some may have had no formal training at all. However, the issue is complicated. Electronic technicians spend years of apprenticeship and practice. Perhaps knowledge of formal physics provides a more direct way to develop understanding early in learning. Just because formal physics may not show up explicitly in skilled performance does not mean it would not facilitate learning.

My interest in human-computer interactions focuses on how much understanding is required to become a skilled user. Clearly, understanding must be defined relative to the task or set of tasks the user wants to perform. Someone who is using a text editor to write a paper needs to understand commands at a very different level than someone who intends to modify the editor. To the person who uses the editor only in order to write a paper, a detailed understanding of the principles of systems design would probably have little beneficial effect; it may even be harmful if the ideas distracted the user.

We must also consider tasks other than the immediate ones the user will perform: users also do experiments; they make mistakes, at times the system fails. For a user to be able to interpret the effects of an experiment or the nature of a mistake requires additional understanding. Even more understanding is required to distinguish between the behavior of a functioning system and a nonfunctioning system. Furthermore, with the increasing proliferation of computers, most users need to learn several systems. We must be concerned with identifying the kinds of understanding a user needs to transfer skills between systems.

In the next two sections I focus on a framework for characterizing user understanding and discuss the role of understanding in performance and learning. I propose that (a) the level at which a user interacts with a device is determined by the tasks being performed; (b) the functions and structures that are understood differ from level to level (see Miyake, in press); (c) a uniform set of criteria are appropriate for evaluating understanding at any level.

My discussion of understanding is informal, focusing primarily on examples from human-computer interfaces to illustrate the various points. Cognitive theories are being developed that more accurately specify the nature of understanding. The framework for characterizing understanding used here is based on current theories of problem solving and language understanding. Ideally, analyses of user understanding will reach a similar level of description, maximizing the extent to which work on human-computer interfaces can benefit from, and contribute to, other developing theories.

A FRAMEWORK FOR CHARACTERIZING UNDERSTANDING

I view the user's task in interacting with a complex device as a problem-solving episode. The user constantly sets goals and must plan how to achieve these goals with the available commands. This view is generally consistent with several other current analyses of human-computer interactions (e.g., Card, Moran, & Newell, 1983; Kieras & Polson, 1982; Moran, 1983; Norman, in press).

Figure 1 presents a typical planning episode in the form of a hierarchical goal structure, or *planning net* (cf. VanLehn & Brown, 1980; Sacerdoti, 1977). At the higher levels of the planning net are global goals. Here the person is using a text editor: The current goal is to edit the paper. This, in turn, generates the additional goal to *"transpose two words."* Since this goal does not correspond to an executable action, further goal specification and planning is required. *"Transpose two words"* is broken down into the subgoals *"delete word1"* and *"insert word1 after word2"* which correspond to the actions of typing *"dw"* (delete word) and *"p"* (put), respectively.

Planning does not necessarily stop with the selection of the primary actions. Associated with actions are requisite conditions that must be taken into account in the planning process:

- *Prerequisites* are conditions that must be satisfied before an action can be performed. The prerequisite of "dw" and "p" is that the cursor be at the appropriate location. Additional goals must be generated to ensure that these prerequisites are satisfied.

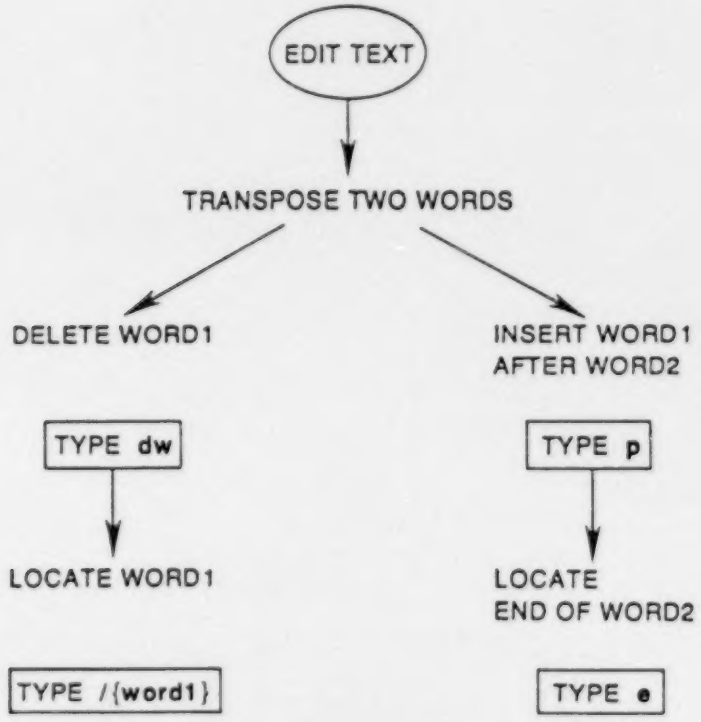

FIGURE 1
Planning net for the task *Transpose two words* (from Riley & O'Malley, 1984).

- *Consequences* are the changes that result from performing an action. In this example, the consequence of "dw" is that the word is deleted from the text. The consequence of "p" is to put the most recently deleted word at the location of the cursor. These consequences define the *order* in which "dw" and "p" must be executed and, furthermore, place restrictions on interleaving plans.

- *Postrequisites*, relevant to some commands, are conditions that must be satisfied after performing an action. For example, the action of inserting text must be followed by pressing the ESCAPE key, to return to command mode.

It is one thing to solve a problem, but quite another to solve it with "understanding." As Greeno (1977, 1978) suggests,[1] understanding depends on three important criteria for evaluating the representation generated during problem solving:

Internal Coherence.
Are the components of the user's representation mutually coherent?
Validity.
Does the representation accurately reflect the behavior of the system?
Integration.
To what extent is the user's representation integrated with the user's knowledge of other areas?

Of course, devices can be described at different levels. Devices can be hierarchically decomposed into structures with each structure serving one or more function at a level of the hierarchy (cf. Brown, Burton, & de Kleer, 1983; Miyake, in press). For example, at one level of description, we can talk about editors, mail systems, and directories; at another level we can talk about the underlying programs, at yet another level we can talk about shuffling bits. The objects differ at different levels, as do the commands and procedures available for operating on those objects. Thus, depending on the particular task a user wishes to accomplish, the objects in the user's problem representation will differ. Nevertheless we can use the same criteria of understanding to evaluate a user's problem representation.

Furthermore, understanding continues to be important regardless of the amount of experience. True, skilled performance of routine tasks probably involves having a large store of automated procedures for accomplishing frequent goals, but not all of a user's interactions with a computer are routine. Among other things, users experiment, make mistakes, and may also have to regenerate procedures that were once automated but forgotten through disuse. For these reasons, a user needs to have more knowledge about a system than simply a list of procedures for accomplishing specific tasks.

WHAT KIND OF UNDERSTANDING DOES A USER NEED?

In this section I discuss in more detail the three criteria for understanding and their roles in performance and learning.

Internal Coherence

Internal Coherence is the extent to which components of knowledge are related in an integrated structure. I argue that a coherent knowledge base facilitates learning and increases the likelihood that commands will be remembered or can be regenerated.

There are many kinds of knowledge that contribute to internal coherence:

[1] Greeno used the terms "coherence," "correspondence," and "correctedness."

BEST COPY AVAILABLE

- Knowledge about the action structure of a command;
- Knowledge about the syntactic structure of a command;
- Knowledge about how a command works;
- Knowledge about objects (e.g., files, programs, buffers);

The knowledge of the action structure of commands is *coherent* in that goals are associated with a command's requisite conditions and component actions. Furthermore, during planning, this knowledge can be used to generate a coherent representation of the hierarchical relations between related sequences of commands. In Figure 1, there is a central high-level goal, "transpose two words," and commands are related to that goal as either primary or enabling goals.

Improved performance (i.e., increased flexibility and efficiency) could be achieved by memorizing additional goal-command pairs. But still, important components of understanding would be missing—components that could have an important influence on performance and learning. Improved understanding could be achieved with the addition of knowledge about the *syntactic structure* of commands, as shown in Figure 2.

In the particular editor used in this example (Berkeley UNIX *vi*), command sequences are defined as the cross-product of an action and a text object: "dw" is the cross-product of the action "d" (for delete) and the object "w" (for word). Furthermore, the text object can be preceded by a number, e.g., "d4w" deletes four words. Other rules involve systematic changes in the scope or direction of a command. Whatever the form of these rules, they add *coherence* to the knowledge, facilitating learning and increasing the likelihood that commands will be remembered or can be regenerated.

> *Note that the coherence of a user's representation depends largely on the system itself. If there is no high-level rule to describe the syntactic structure of a command language, the user is restricted in forming a coherent representation of the relation between command sequences. Indeed, several studies (Payne & Green, 1983; Reisner, 1981) have linked the degree of syntactic coherence to users' ability to learn, remember, and regenerate commands.*

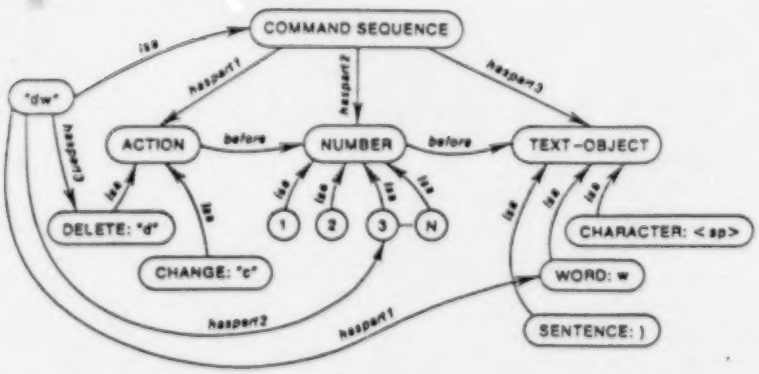

FIGURE 2
Knowledge about the syntactic structure of commands.

AVAILABLE

Knowledge about how a command works also adds internal coherence to the user's representation of the editor command structure. For example, in Figure 1, the consequence of deleting a word is not related directly to the action of putting the word in a new location. That is, there is no explicit representation of the relation between the consequence of "dw" and the prerequisite of "p." In fact, the consequence of "dw" is to place the word in a temporary storage place, or buffer. The prerequisite of "p" is that there be something in the buffer and its consequence is to put the buffer contents at the location of the cursor. Figure 3 shows how this additional knowledge adds coherence to the user's representation by explicitly identifying the relation between the consequence of "dw" and the prerequisite of "p."

Other kinds of knowledge also contribute to coherence. Most editors keep the modified version of the text in active memory and do not update the permanent copy on the user's disk until the user leaves the editor. In order to understand this fact, the user has to know that the editor is a program and be familiar with concepts of memory and disk storage (see Kieras & Polson, 1982, for further discussion). Users also need to know about the properties of the objects that are operated on by these programs and commands. A coherent representation of why the command to edit a file is sometimes preceeded by a command to change directories and/or a command to change the file protection depends upon the user knowing that files are organized in directories (which are also files) and that each file has a protection status that is checked by the editor program. A coherent representation of the effects of editor commands requires knowledge of how text objects are specified, e.g., the different notions of a "line" discussed by Owen (in press).

An interesting issue concerns whether there is a tradeoff between internal coherence and learnability. Coming up with a single coherent view of a system may involve structural models and rules that are quite difficult to learn. Multiple "distributed" models may be less coherent

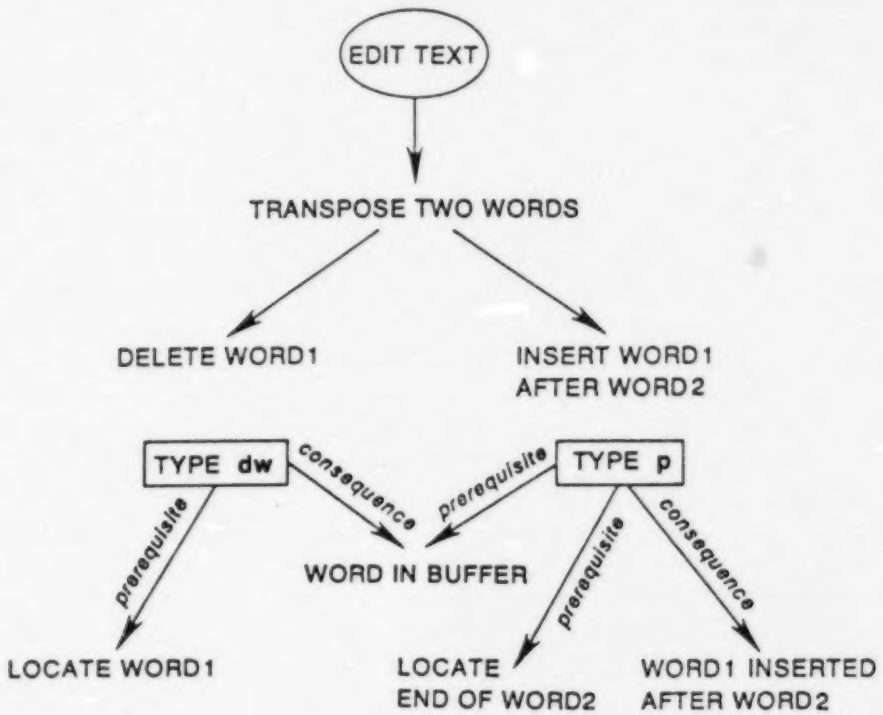

FIGURE 3
Knowledge about buffers.

overall but may be easier to learn (see diSessa's, in press, discussion of this point).

Another issue concerns whether knowledge at any level of description increases coherence. Would an explanation of the editor in elaborate programming terms or design terms increase coherence? Or would it simply provide an alternative perspective (see Miyake, in press), without directly affecting the connections between the objects of concern? That is, this additional knowledge about underlying mechanisms and design constraints may add more nodes and links to the knowledge representation, but it is not clear that the density (i.e., coherence) of the representation would increase.

Several studies (e.g., Halasz & Moran, 1983; Kieras & Bovair, 1984) show that having some model of how a device works facilitates learning, retention and/or invention of procedures for operating a device. However, the benefits of a device model depend on whether it allows the user to infer the exact steps required to operate the device. Specifically, inferring procedures requires information about the system topology (what is connected to what) and the principle of flow of control. Thus the critical "how-it-works" information is the specific description of the controls and their path relations to the internal components. Neither details about the nature of the components, nor general principles about how a system works enable users to infer procedures. How-it-works information must be selected so that it is actually relevant to the user's task—it must explain how or why a goal must be accomplished.

Notice that this does not constitute a general argument against the role of formal principles in learning complex skills. In basic electronics, physics principles can be used to constrain the quantities in the problem representation, adding coherence. Physics principles also provide a way of relating different procedures, thereby embedding them in a higher-order organized structure that could be beneficial for retention and application. (See Riley, 1984, for a detailed discussion of the role of formal principles in learning basic electronics.)

Validity

Validity is the extent to which the user's components of knowledge are consistent with the behavior of the system. To the extent that the user's knowledge corresponds to the system's behavior, the user will be able to *explain* and *predict* the effects of commands and *generate* new sequences of commands to achieve desired effects. I identify different kinds of validity that are roughly equivalent to diSessa's distinction between functional models, distributed models, and structural models (diSessa, in press).

Distributed and structural models correspond to how-it-works knowledge at varying levels of description. How-it-works knowledge enables users to predict the behavior of any sequence of commands, justify rules when they are correct, explain their limitations, and go beyond them. However, as diSessa (in press) and Young (1981, 1983) point out, structural models are not always sufficient to enable fluent interaction with the system. A significant amount of problem solving may be required to invent a way to achieve a goal. It seems highly unlikely that users would continually rederive procedures through explicit reference to models and rules. For example, when transposing two letters for the 100th time it is unlikely that the user explicitly represents the letter going into the buffer and then being moved from the buffer to a new location in the text. Rather, the keystrokes corresponding to this procedure are probably done automatically, as a unit (cf. Robertson & Black, 1983).

> *Note validity does not depend on whether the user's models are identical to (contain same objects and relations as) the Design Model, as long as they lead to the same predictions. At the same time, if correspondence is maintained as a user's knowledge encompasses more and more of the system, the space of functionally equivalent models probably decreases.*

Users often generate context-dependent validity to achieve coherence at the expense of accuracy. Lewis (in press) shows how users generate coherent explanations of a command's function in a specific context but these explanations often do not predict the command's function in

another context (see de Kleer & Brown 1983). Robust models contain no implicit, context-specific assumptions about a command's function. Such models should predict the command's effects and functions, regardless of the particular command string it is embedded in. Robust models correspond to the behavior of the system for sequences of commands that were unanticipated and therefore not precompiled.

Consider robustness in the context of the buffer model. Knowledge about which commands change the contents of the buffer and which commands access the contents of the buffer suffices to predict and explain the effects of many command sequences. For example, the command for changing a word, "cw," also has the consequence of replacing the contents of the buffer with the word that was changed (and therefore deleted). Following a "cw" event with the command "p" has the effect of putting the changed word at the current location of the cursor. If "dw" were executed between "cw" and "p," the model correctly predicts that consequence of "p" would be different.

An interesting issue concerns the degree of robustness acquired with experience. I use diSessa's terms "structural" and "distributed" to characterize two different perspectives. de Kleer and Brown (1983) suggest that with experience, users acquire increasingly robust device knowledge--approaching what diSessa refers to as "structural models." Initial component models invariably include many implicit assumptions about the overall functioning of the device, which may or may not be correct. With experience, component models become more robust by making implicit assumptions explicit. This transition is primarily motivated by discovering violations of consistency, validity, and/or robustness constraints. That is, the causal model may contain conflicting models for a single component type, the causal model may not correspond to the observed behavior of the device, or the causal model may only correspond when the device is functioning correctly. Discoveries like these encourage the learner to identify an underlying implicit assumption in a component model, gradually making the component models more robust. These assumptions (and future ones) need not be discarded—they lead to very efficient reasoning about the correctly functioning device from which they originated. Thus with experience, the learner probably stores more and more device-specific assumptions along with the component models. These new component models are robust in that, if assumptions are violated, the learner can automatically distinguish these assumptions from the actual model and proceed to envision a correct causal model of the device.

I extrapolate from diSessa's notion of distributed models to propose an alternative perspective. It is possible that users continue to have many models to account for different aspects of the system and no single model is robust (in the strong sense used by de Kleer and Brown). Does an expert electrician ever acquire robust models of the circuits being tested? Does an automechanic ever acquire a robust model of a car engine? What about operators of steam plants and nuclear power plants? Often there may be no single model that is perfect for describing the behavior of the system (cf. Bott, 1978; Rumelhart & Norman, 1980). The distributed perspective says that improvements in skill result from learning to apply the right model at the right time and perhaps refining models with context-specific knowledge.

It is likely that the distributed view of learning describes most users. For one thing, inferring a robust causal model is difficult, if not impossible, even for relatively simple devices (cf. Miyake, in press; Young, 1981). Young's (1981) account of how he derived a robust model of the stack calculator shows the importance of systematically generating, testing, and revising hypotheses. In contrast, many users probably form hypotheses on the fly on the basis of isolated examples, fail to experiment systematically, or may be afraid to experiment (see Lewis, in press, for further discussion). Especially when a detailed model is not required to operate a device, users probably do not take the time to generate one. One of Miyake's subjects had used a sewing machine for years without generating a detailed model of how the machine worked—although when forced, she generated and refined a very robust model.

There are several reasons why understanding more about the kinds of models users generate is interesting from a design perspective. Knowing more about the kinds of models people develop with experience would tell us something about the limits on the complexity of the models with which humans reason—or at least the kinds of models with which they prefer to

reason. It would also provide us with information about how multiple mental models are coordinated in learning and performance. Furthermore, information about how those models are acquired, modified, abandoned, may be useful in guiding users through progressive layers of competency.

Integration

Integration is the extent to which the components of knowledge in one domain are tied to other components of a person's knowledge. Ease of learning depends on the extent that objects and relations in a new domain can be connected to familiar components of knowledge.

One kind of integration that is especially important for new users is integration with general knowledge. For example, many editor goals are *connected* with goals familiar to the user—changing and deleting characters and words, inserting new text, correspond to general goals in editing text, regardless of whether the text is handwritten, typewritten, or written using a computer text editor.

> *However, the semantics of the actions are not necessarily the same. For example, on a typewriter, the actions associated with Change-word involve erasing the old word, making space for the new word, and then typing in the new word. The goals "Transpose-Two-Letters" and "Repeat-Last-Command" have no direct counterpart in typewriting or handwriting (but transpose-two-words is used in proof-reading; also, these actions are frequently carried out in several steps, e.g., with white-out). See Owen's discussion of the same issue (in press).*

A second kind of integration is with other knowledge of the other systems. Buffers and cross-product rules are not specific to this particular text editor or to text editors in general. At the right level of description, these concepts can be *connected* to other systems, leading to efficient transfer of knowledge. For example, most editors use buffers to store text that has been recently deleted or inserted. Even though editors may differ in the specific ways they use buffers, simply knowing that buffers are used constrains hypotheses, and explanations.

Integration is not necessarily beneficial. As Lewis points out (in press) learners unfamiliar with a domain often make connections that are not valid, resulting in inefficient performance and errors.[2] Users learning to use a text editor for the first time connect commands for inserting text to their knowledge of inserting text using a typewriter (or to their knowledge of inserting text using paper and pencil). As a result, users often think that space has to be made before text can be inserted. Similarly, users think that any text visible on the screen is in the file and, vice versa, any text not visible on the screen is not in the file. This leads to predictable confusion in editors where inserting text has the consequence of typing over existing text until a special key is pressed to terminate the input mode. Subjects are likely to think that the over-typed text is no longer in the file. Predictable confusion also results in editors that leave deleted text on the screen during input mode, even though the user has backspaced over the text to delete it. In this case the user is likely to think that the text has not been deleted, when in fact it has been.

A major problem with naive models is that they are surprisingly persistent (cf. Clement, 1983; DiSessa, 1983). Among the frequently suggested reasons for the persistence of these naive models are that (a) students may misinterpret or distort information to fit their naive views; (b) students may have several models for different instances of the same phenomenon and shift between models to interpret the various situations; (c) students may focus only on the salient aspects of an event and ignore less salient (or invisible) factors. Clearly, further empirical and theoretical analyses are required to identify the cognitive processes that could lead to necessary changes in user's models.

---

[2] (cf. Bott, 1978; Douglas & Moran, 1983; Halasz & Moran, 1982; Lewis & Mack, 1982; Riley & O'Malley, 1984; Rumelhart & Norman, 1980).

SUMMARY

Understanding facilitates learning, provides predictive and explanatory power, increases the likelihood that procedures will be remembered or can be regenerated, and enables the transfer of skills. In unfamiliar situations, understanding improves the efficiency, flexibility, and reliability of performance, permits and constrains generation of new procedures, and facilitates checking answers.

A direct outcome of the analyses presented in this chapter is a view of understanding as a multi-dimensional quality rather than as something one has or one does not have. Understanding is related to three characteristics of the user's knowledge: internal coherence, validity, and integration. Coherence concerns the degree to which the user's components of knowledge are related in an integrated structure. Validity concerns the extent to which the user's components of knowledge accurately reflect the behavior of the system. Integration concerns the degree to which the components of knowledge are related to other components of user's knowledge.

The degree of internal coherence, validity, and integration does not depend on a single aspect of knowledge, but upon several. This emphasizes that a user should not be considered as either performing with or without understanding, since it clearly is possible for the user to have acquired some components of knowledge and not others.

REFERENCES

Bott, R. A. (1978). *A study of complex learning, theory and methodologies.* Unpublished doctoral dissertation, University of California, San Diego.

Brown, J. S., Burton, R. R., & deKleer, J. (1983). Pedagogical, natural language, and knowledge engineering techniques in SOPHIE I, II, and III. In D. Sleeman & J. S. Brown (Eds.), *Intelligent tutoring systems.* New York: Academic Press.

Card, S. K., Moran, T. P., & Newell, A. (1983). *The psychology of human-computer interaction.* Hillsdale, NJ: Erlbaum.

Clement, J. (1983). A conceptual model discussed by Galileo and used intuitively by physics students. In D. Gentner & A. L. Stevens (Eds.), *Mental models.* Hillsdale, NJ: Erlbaum.

deKleer, J., & Brown, J. S. (1983). Assumptions and ambiguities in mechanistic mental models. In D. Gentner & A. L. Stevens (Eds.), *Mental models.* Hillsdale, NJ: Erlbaum.

diSessa, A. A. (1983). Phenomenology and the evolution of intuition. In D. Gentner & A. L. Stevens (Eds.), *Mental models.* Hillsdale, NJ: Erlbaum.

diSessa, A. A. (in press). Models of computation. In D. A. Norman & S. W. Draper (Eds.), *User centered system design: New perspectives in human-machine interaction.* Hillsdale, NJ: Erlbaum.

Douglas, S. A., & Moran, T. P. (1983). Learning text editor semantics by analogy. In A. Janda (Ed.), *Proceedings of the CHI '83 Conference on Human Factors in Computing Systems* (pp. 207-211). New York: ACM.

Gentner, D., & Gentner, D. R. (1983). Flowing waters or teaming crowds: Mental models of electricity. In D. Gentner & A. L. Stevens (Eds.), *Mental models.* Hillsdale, NJ: Erlbaum.

Greeno, J. G. (1977). Process of understanding in problem solving. In N. J. Castellan, D. B. Pisoni, & G. R. Potts (Eds.), *Cognitive theory* (Vol. 2, pp. 43-83). Hillsdale, NJ: Erbaum.

Greeno, J. G. (1978). Understanding and procedural knowledge in mathematics instruction. *Educational Psychologist, 12*, 262-283.

Halasz, F. G., & Moran, T. P. (1982). Analogy considered harmful. *Proceeding of the Conference on Human Factors in Computer Systems* (pp. 383-386). New York: ACM. Gaithersburg, MD.

Halasz, F. G., & Moran, T. P. (1983). Mental models and problem solving in using a calculator. Pp. 212-216. In A. Janda (Ed.), *Proceedings of the CHI '83 Conference on Human Factors in Computing Systems* (pp. 212-216). New York: ACM.

Kieras, D. E., & Bovair, S. (1984). The role of a mental model in learning to operate a device. *Cognitive Science, 8*, 255 - 273.

Kieras, D. E., & Polson, P. G. (1983). A generalized transition network representation for interactive systems. In A. Janda (Ed.), *Proceedings of the CHI '83 Conference on Human Factors in Computing Systems* (pp. 103-106). New York: ACM.

Kieras, D. E., & Polson, P. G. (in press). An approach to the formal analysis of user complexity. *International Journal of Man-Machine Studies.*

Lewis, C. (in press). Understanding what's happening in system interactions. In D. A. Norman & S. W. Draper (Eds.), *User centered system design: New perspectives in human-machine interaction*. Hillsdale, NJ: Erlbaum.

Lewis, C. H., & Mack, R. (1982). Learning to use a text processing system: Evidence from "thinking aloud" protocols. *Proceedings of the Conference on Human Factors in Computer Systems* (pp. 387-392). New York: ACM.

Miyake, N. (in press). Constructive interaction.

Moran, T. P. (1983). Getting into a system: External-internal task mapping analysis. Pp. 45-49. In A. Janda (Ed.), *Proceedings of the CHI '83 Conference on Human Factors in Computing Systems* (pp. 45-49). New York: ACM.

Norman, D. A. (in press). Cognitive engineering. In D. A. Norman & S. W. Draper (Eds.), *User centered system design: New perspectives in human-machine interaction*. Hillsdale, NJ: Erlbaum.

Norman, D. A., Rumelhart, D. E., & the LNR Research Group. (1975). *Explorations in cognition*. San Francisco: Freeman.

Owen, D. (in press). Naive theories of computation. In D. A. Norman & S. W. Draper (Eds.), *User centered system design: New perspectives in human-machine interaction*. Hillsdale, NJ: Erlbaum.

Payne, S. J., & Green, T. R. G. (1983). The user's perception of the interaction language: A two-level model. In A. Janda (Ed.), *Proceedings of the CHI '83 Conference on Human Factors in Computing Systems*. (pp. 202-206). New York: ACM.

Riley, M. S. (1984). *Structural understanding in performance and learning*. Unpublished doctoral dissertation, University of Pittsburgh.

Roberts, T. L., & Moran, T. P. (1983). The evaluation of text editors: Methodology and

empirical results. *Communications of the ACM, 26,* 265-283.

Robertson, S. P., & Black, J. B. (December, 1983). Planning units in text editing behavior. Pp. 217-221. In A. Janda (Ed.), *Proceedings of the CHI '83 Conference on Human Factors in Computing Systems* (pp. 217-221). New York: ACM.

Reisner, P. (1981). Formal grammar and human factors design of an interactive graphics system. *IEEE Transactions on Software Engineering, 5,* 229-240.

Rumelhart, D. E., & Norman, D. A. (1981). *Analogical processes in learning.* In J. R. Anderson (Ed.), *Cognitive skills and their acquistion.* Hillsdale, NJ: Erlbaum.

Sacerdoti, E. D. (1977). *A structure for plans and behavior.* New York: Elsevier.

VanLehn, K., & Brown, J. S. (1980). Planning nets: A representation for formalizing analogies and semantic models of procedural skills. In R. E. Snow, P. A. Federico, & W. E. Montague (Eds.), *Aptitude, learning, and instruction: Vol. 2. Cognitive process analyses of learning and problem solving.* Hillsdale, NJ: Erlbaum.

Young, R. M. (1981). The machine inside the machine: Users' models of pocket calculators. *International Journal of Man-Machine Studies, 15,* 51-85.

Young, R. M. (1983). Surrogates and mappings: Two kinds of conceptual models for interactive devices. In D. Gentner & A. Stevens (Eds.), *Mental models.* Hillsdale, NJ: Erlbaum.

ICS Technical Report List

The following is a list of publications by people in the Institute for Cognitive Science. For reprints, write or call:

Institute for Cognitive Science, C-015
University of California, San Diego
La Jolla, CA 92093
(619) 452-6771

8301. David Zipser. *The Representation of Location.* May 1983.

8302. Jeffrey Elman and Jay McClelland. *Speech Perception as a Cognitive Process: The Interactive Activation Model.* April 1983. Also published in N. Lass (Ed.), *Speech and language: Volume 10*, New York: Academic Press, 1983.

8303. Ron Williams. *Unit Activation Rules for Cognitive Networks.* November 1983.

8304. David Zipser. *The Representation of Maps.* November 1983.

8305. The HMI Project. *User Centered System Design: Part I, Papers for the CHI '83 Conference on Human Factors in Computer Systems.* November 1983. Also published in A. Janda (Ed.), *Proceedings of the CHI '83 Conference on Human Factors in Computing Systems.* New York: ACM, 1983.

8306. Paul Smolensky. *Harmony Theory: A Mathematical Framework for Stochastic Parallel Processing.* December 1983. Also published in *Proceedings of the National Conference on Artificial Intelligence, AAAI-83*, Washington DC, 1983.

8401. Stephen W. Draper and Donald A. Norman. *Software Engineering for User Interfaces.* January 1984. Also published in *Proceedings of the Seventh International Conference on Software Engineering*, Orlando, FL, 1984.

8402. The UCSD HMI Project. *User Centered System Design: Part II, Collected Papers.* March 1984. Also published individually as follows: Norman, D.A. (in press), Stages and levels in human-machine interaction, *International Journal of Man-Machine Studies*; Draper, S.W., The nature of expertise in UNIX; Owen, D., Users in the real world; O'Malley, C., Draper, S.W., & Riley, M., Constructive interaction: A method for studying user-computer-user interaction; Smolensky, P., Monty, M.L., & Conway, E., Formalizing task descriptions for command specification and documentation; Bannon, L.J., & O'Malley, C., Problems in evaluation of human-computer

BEST COPY AVAILABLE

interfaces: A case study; Riley, M., & O'Malley, C., Planning nets: A framework for analyzing user-computer interactions; all published in B. Shackel (Ed.), *INTERACT '84, First Conference on Human-Computer Interaction*, Amsterdam: North-Holland, 1984; Norman, D.A., & Draper, S.W., Software engineering for user interfaces, *Proceedings of the Seventh International Conference on Software Engineering*, Orlando, FL, 1984.

8403. Steven L. Greenspan and Eric M. Segal. *Reference Comprehension: A Topic-Comment Analysis of Sentence-Picture Verification.* April 1984. Also published in *Cognitive Psychology, 16*, 556-606, 1984.

8404. Paul Smolensky and Mary S. Riley. *Harmony Theory: Problem Solving, Parallel Cognitive Models, and Thermal Physics.* April 1984. The first two papers are published in *Proceedings of the Sixth Annual Meeting of the Cognitive Science Society*, Boulder, CO, 1984.

8405. David Zipser. *A Computational Model of Hippocampus Place-Fields.* April 1984.

8406. Michael C. Mozer. *Inductive Information Retrieval Using Parallel Distributed Computation.* May 1984.

8407. David E. Rumelhart and David Zipser. *Feature Discovery by Competitive Learning.* July 1984. Also published in *Cognitive Science, 9*, 75-112, 1985.

8408. David Zipser. *A Theoretical Model of Hippocampal Learning During Classical Conditioning.* December 1984.

8501. Ronald J. Williams. *Feature Discovery Through Error-Correction Learning.* May 1985.

8502. Ronald J. Williams. *Inference of Spatial Relations by Self-Organizing Networks.* May 1985.

8503. Edwin L. Hutchins, James D. Hollan, and Donald A. Norman. *Direct Manipulation Interfaces.* May 1985. To be published in D. A. Norman & S. W. Draper (Eds.), *User Centered System Design: New Perspectives in Human-Computer Interaction.* Hillsdale, NJ: Erlbaum.

8504. Mary S. Riley. *User Understanding.* May 1985. To be published in D. A. Norman & S. W. Draper (Eds.), *User Centered System Design: New Perspectives in Human-Computer Interaction.* Hillsdale, NJ: Erlbaum.

Earlier Reports by People in the Cognitive Science Lab

The following is a list of publications by people in the Cognitive Science Lab and the Institute for Cognitive Science. For reprints, write or call:

Institute for Cognitive Science, C-015
University of California, San Diego
La Jolla, CA 92093
(619) 452-6771

ONR-8001. Donald R. Gentner, Jonathan Grudin, and Eileen Conway. *Finger Movements in Transcription Typing*. May 1980.

ONR-8002. James L. McClelland and David E. Rumelhart. *An Interactive Activation Model of the Effect of Context in Perception: Part I*. May 1980. Also published in *Psychological Review, 88,5*, pp. 375-401, 1981.

ONR-8003. David E. Rumelhart and James L. McClelland. *An Interactive Activation Model of the Effect of Context in Perception: Part II*. July 1980. Also published in *Psychological Review, 89, 1*, pp. 60-94, 1982.

ONR-8004. Donald A. Norman. *Errors in Human Performance*. August 1980.

ONR-8005. David E. Rumelhart and Donald A. Norman. *Analogical Processes in Learning*. September 1980. Also published in J. R. Anderson (Ed.), *Cognitive skills and their acquisition*. Hillsdale, NJ: Erlbaum, 1981.

ONR-8006. Donald A. Norman and Tim Shallice. *Attention to Action: Willed and Automatic Control of Behavior*. December 1980.

ONR-8101. David E. Rumelhart. *Understanding Understanding*. January 1981.

ONR-8102. David E. Rumelhart and Donald A. Norman. *Simulating a Skilled Typist: A Study of Skilled Cognitive-Motor Performance*. May 1981. Also published in *Cognitive Science, 6*, pp. 1-36, 1982.

ONR-8103. Donald R. Gentner. *Skilled Finger Movements in Typing*. July 1981.

ONR-8104. Michael I. Jordan. *The Timing of Endpoints in Movement*. November 1981.

ONR-8105. Gary Perlman. *Two Papers in Cognitive Engineering: The Design of an Interface to a Programming System and MENUNIX: A Menu-Based Interface to UNIX (User Manual)*. November 1981. Also published in *Proceedings of the 1982 USENIX Conference*, San Diego, CA, 1982.

ONR-8106. Donald A. Norman and Diane Fisher. *Why Alphabetic Keyboards Are Not Easy to Use: Keyboard Layout Doesn't Much Matter*. November 1981. Also published in *Human Factors, 24*, pp. 509-515, 1982.

ONR-8107. Donald R. Gentner. *Evidence Against a Central Control Model of Timing in Typing*. December 1981. Also published in *Journal of Experimental Psychology: Human Perception and Performance, 8*, pp. 793-810, 1982.

ONR-8201. Jonathan T. Grudin and Serge Larochelle. *Digraph Frequency Effects in Skilled Typing*. February 1982.

ONR-8202. Jonathan T. Grudin. *Central Control of Timing in Skilled Typing*. February 1982.

ONR-8203. Amy Geoffroy and Donald A. Norman. *Ease of Tapping the Fingers in a Sequence Depends on the Mental Encoding*. March 1982.

ONR-8204. LNR Research Group. *Studies of Typing from the LNR Research Group: The role of context, differences in skill level, errors, hand movements, and a computer simulation.* May 1982. Also published in W. E. Cooper (Ed.), *Cognitive aspects of skilled typewriting*. New York: Springer-Verlag, 1983.

ONR-8205. Donald A. Norman. *Five Papers on Human-Machine Interaction.* May 1982. Also published individually as follows: Some observations on mental models, in D. Gentner and A. Stevens (Eds.), *Mental models*, Hillsdale, NJ: Erlbaum, 1983; A psychologist views human processing: Human errors and other phenomena suggest processing mechanisms, in *Proceedings of the International Joint Conference on Artificial Intelligence*, Vancouver, 1981; Steps toward a cognitive engineering: Design rules based on analyses of human error, in *Proceedings of the Conference on Human Factors in Computer Systems*, Gaithersburg, MD, 1982; The trouble with UNIX, in *Datamation, 27,12*, November 1981, pp. 139-150; The trouble with networks, in *Datamation*, January 1982, pp. 188-192.

ONR-8206. Naomi Miyake. *Constructive Interaction.* June 1982.

ONR-8207. Donald R. Gentner. *The Development of Typewriting Skill.* September 1982. Also published as Acquisition of typewriting skill, in *Acta Psychologica, 54*, pp. 233-248, 1983.

ONR-8208. Gary Perlman. *Natural Artificial Languages: Low-Level Processes.* December 1982. Also published in *The International Journal of Man-Machine Studies, 20*, pp. 373-419, 1984.

ONR-8301. Michael C. Mozer. *Letter Migration in Word Perception.* April 1983. Also published in *Journal of Experimental Psychology: Human Perception and Performance, 9, 4*, pp. 531-546, 1983.

ONR-8302. David E. Rumelhart and Donald A. Norman. *Representation in Memory*. June 1983. To appear in R. C. Atkinson, G. Lindzey, & R. D. Luce (Eds.), *Handbook of experimental psychology*. New York: Wiley (in press).